This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Previously Presented) A method in a computer system for performing a transaction that includes multiple requests, the method comprising:

an application interface of a client computer receiving a request from an application program;

the application interface associating a transaction identifier with the request, wherein the transaction identifier identifies a transaction that the request is associated with;

the application interface forming a message including the transaction identifier and the request;

the application interface sending the message to a server, comprising the steps of opening a connection with the server, sending the message over the connection, initializing a timer upon receipt of a reply from the server, determining whether a timeout period has expired before another request is received from the application program, and closing the connection when the timeout period has expired;

the server receiving the message; and

the server processing the request in a context of the transaction identified by the transaction identifier in the message.

2. (Original) The method as claimed in claim 1, further comprising:

the application interface associating a sequence indicator to the request, wherein the sequence indicator indicates in what sequence the server should process the request within the context of the transaction; and

wherein forming the message comprises including the sequence indicator in the message.

3. (Canceled)

4. (Original) The method as claimed in claim 1, further comprising:

receiving a reply from the server in response to the request;

determining that the reply includes a redirect request that indicates that the request should be sent to another server; and

sending the message to the another server.

5. (Original) The method as claimed in claim 1, further comprising:

the server allocating a database connection to the transaction; and

the server processing the requests that form a part of the transaction over the database connection allocated to the transaction.

6. (Original) The method as claimed in claim 5, further comprising:

the application interface including a sequence indicator in the message, wherein the sequence indicator indicates in what sequence the server should process the request within the context of the transaction; and

wherein processing the requests comprises processing the requests in an order indicated by the sequence indicator.

7. (Previously Presented) A method in a computer system for a client to perform multiple requests in the context of a transaction, the method performed by an application interface associated with the client, the method comprising:

receiving a request from an application program;

associating a transaction identifier with the request, wherein the transaction identifier identifies a transaction that the request is associated with;

forming a message including the transaction identifier and the request; and

sending the message to a server, comprising the steps of opening a connection with the server, sending the message over the connection, initializing a timer upon receipt of a reply from the server, determining whether a timeout period has expired before another request is received from the application program, and closing the connection when the timeout period has expired.

8. (Original) The method as claimed in claim 7, further comprising assigning the transaction identifier to the transaction if the request is a request to open a new transaction.

9. (Original) The method as claimed in claim 7, further comprising:

associating a sequence indicator with the request, wherein the sequence indicator indicates in what sequence the server should process the request within the context of the transaction; and

wherein forming the message comprises including the sequence indicator within the message.

10. (Original) The method as claimed in claim 7, further comprising:

multiplexing together multiple requests that are destined for the server; and

sending the multiple requests to the server in a multiplexed format.

11. (Original) The method as claimed in claim 7, further comprising:

receiving a reply from the server in response to sending the message; and

sending the reply to the application program.

12. (Previously Presented) A method in a computer system for a client to redirect a request from a first server to a second server, the method performed by an application interface, the method comprising:

receiving a request from an application program, wherein the application interface is separate from the application program;

formatting the request into a message;

sending the message to the first server;

receiving a reply from the first server in response to the message, wherein the reply includes a redirect request that indicates that the second server is a correct destination for the message; and

sending the message to the second server without involving the application program.

13. (Previously Presented) The method as claimed in claim 12, wherein sending the message to the first server comprises:

opening a connection with the server;

sending the message over the connection;

initializing a timer upon receipt of a reply from the server;

determining whether a timeout period has expired before another request is received from the application program; and

closing the connection when the timeout period has expired.

14. (Original) The method as claimed in claim 12, further comprising the application interface storing the redirect request so that subsequent similar requests can be initially directed to the second server rather than to the first server.

15. (Currently Amended) A method in a computer system for a client to send one or more messages to a server, the method comprising:

opening a connection with the server;

sending the one or more message over the connection;

initializing a timer upon receipt of a reply from the server;

determining whether a timeout period has expired before another request is received from ~~the~~ an application program of the client; and

closing the connection when the timeout period has expired.

16. (Previously Presented) The method as claimed in claim 15, wherein determining whether the timeout period has expired comprises comparing a value of the timer to the timeout period.

17. (Original) The method as claimed in claim 15, further comprising:

receiving another request destined for the server within the timeout period; and

sending the another request to the server over the connection.

18. (Original) The method as claimed in claim 17, further comprising:

re-initializing and re-starting the timer when a reply corresponding to the another request is received from the server.

19. (Previously Presented) A method in a computer system for a server to manage transactions comprising:

receiving a message from a client, wherein the message includes a transaction identifier that indicates that a request specified in the message should be performed in a context of a transaction;

processing the request in the context of the transaction identified by the transaction identifier;

reserving a database connection for the transaction between the server and a database corresponding to the transaction;

determining whether a free connection to the database is available;

if the free connection is not available, opening a new connection to the database; and

reserving the database connection by mapping the transaction identifier to the new connection.

20. (Original) The method as claimed in claim 19, wherein the message includes a sequence indicator which indicates a sequence, within the transaction, that the request should be processed, and wherein processing the request comprises processing the request in the sequence.

21. (Original) The method as claimed in claim 20, further comprising:

sequencing the request by placing the request, and other requests that comprise the transaction, in a transaction queue associated with the transaction; and

wherein processing the request comprises processing the request in a sequence of requests in the transaction queue.

22. (Canceled)

23. (Previously Presented) The method as claimed in claim 19, further comprising:

if the free connection is available, reserving the database connection by mapping the transaction identifier to the free connection.

24. (Canceled)

25. (Previously Presented) The method as claimed in claim 19, further comprising:

if the free connection is not available, determining whether a maximum number of connections to the database has been reached; and

if the maximum number of connections to the database has been reached, stalling the request until the free connection is available.

26. (Original) The method as claimed in claim 19, further comprising:

closing the connection when the transaction is committed or aborted.

27. (Original) The method as claimed in claim 19, further comprising:

deallocating the connection when the transaction is committed or aborted.

28. (Previously Presented) A computer system for processing requests from application programs, the computer system comprising:

a processor for executing an application interface which receives requests destined for the server from at least one application program, associates transaction identifiers with each of the requests that are associated with transactions, and forms messages including the transaction identifiers and the requests; and

an interface to the server, coupled to the processor through a bus, which sends the messages to the server by opening a connection with the server, sending the messages over the connection, initializing a timer upon receipt of a reply from the server, determining whether a timeout period has expired before a subsequent request is received from the application program, and closing the connection when the timeout period has expired.

29. (Original) The computer system as claimed in claim 28, wherein the processor is further for associating sequence indicators with the requests, wherein the sequence indicators indicate the order that the requests within a transaction should be performed by the server.

30. (Original) A computer system for processing requests from application programs, the computer system comprising:

a processor for executing an application interface which receives a request destined for a first server from an application program, formats the request into a message, sends the message to the first server via a network interface, receives a reply from the first server in response to the message, wherein the reply includes a redirect request that indicates that a second server is a correct destination for the message, and sends the message to the second server via the network interface without involving the application program; and

the network interface, coupled to the processor through a bus, which sends the message to the first server and the second server.

31. (Previously Presented) A computer system for processing requests from application programs, the computer system comprising:

a processor for executing an application interface which receives requests destined for a server from at least one application program, opens a connection with the server, sends the one or more requests over the connection, initializes a timer upon receipt of a reply from the server, determines whether a timeout period has expired before another request is received from the at least one application program, and closes the connection when the timeout period has expired; and

an interface to the server, coupled to the processor through a bus, which sends the one or more requests to the server.

32. (Original) The computer system as claimed in claim 31, wherein the processor is further for sending one or more additional requests destined for the server over the connection during the timeout period.

33. (Previously Presented) A computer system for processing requests received in messages from one or more clients, the computer system comprising:

a processor for executing a transaction manager that receives, via a network interface, a message from a client, wherein the message includes a transaction identifier that indicates that a request specified in the message should be performed in a context of a transaction, and for processing the request in the context of the transaction identified by the transaction identifier, said transaction manager reserving a database connection for the transaction between the server and a database corresponding to the transaction, determining whether a free connection to the database is available, if the free connection is not available opening a new connection to the database, and reserving the database connection by mapping the transaction identifier to the new connection; and

the network interface for receiving the message from the client.

34. (Original) The computer system as claimed in claim 33, wherein the message also includes a sequence indicator, which indicates a sequence, within the transaction, that the request should be processed, and wherein the transaction manager processes the request in the sequence.

35. (Canceled)

36. (Previously Presented) A computer-readable medium holding computer executable instructions, the computer-readable medium for performing a method in a computer system for a client to perform multiple requests in the context of a transaction, the method performed by an application interface associated with the client, the method comprising:

receiving a request from an application program;

associating a transaction identifier with the request, wherein the transaction identifier identifies a transaction that the request is associated with;

forming a message including the transaction identifier and the request; and

sending the message to a server, comprising the steps of opening a connection with the server, sending the message over the connection, initializing a timer upon receipt of a reply from the server, determining whether a timeout period has expired before another request is received from the application program, and closing the connection when the timeout period has expired.

37. (Original) A computer-readable medium holding computer executable instructions, the computer-readable medium for performing a method in a computer system for a client to redirect a request from a first server to a second server, the method performed by an application interface comprising:

receiving a request from an application program, wherein the application interface is separate from the application program;

formatting the request into a message;

sending the message to the first server;

receiving a reply from the first server in response to the message, wherein the reply includes a redirect request that indicates that the second server is a correct destination for the message; and

sending the message to the second server without involving the application program.

38. (Previously Presented) A computer-readable medium holding computer executable instructions, the computer-readable medium for performing a method in a computer system for a client to send one or more messages to a server, the method comprising:

opening a connection with the server;

sending the one or more message over the connection;

initializing a timer upon receipt of a reply from the server;

determining whether a timeout period has expired before another message is received for sending to the server; and

closing the connection when the timeout period has expired.

39. (Previously Presented) A computer-readable medium holding computer executable instructions, the computer-readable medium for performing a method in a computer system for a server to manage transactions comprising:

receiving a message from a client, wherein the message includes a transaction identifier that indicates that a request specified in the message should be performed in a context of a transaction;

processing the request in the context of the transaction identified by the transaction identifier;

reserving a database connection for the transaction between the server and a database corresponding to the transaction;

determining whether a free connection to the database is available;

if the free connection is not available, opening a new connection to the database; and

reserving the database connection by mapping the transaction identifier to the new connection.